Communication Networks

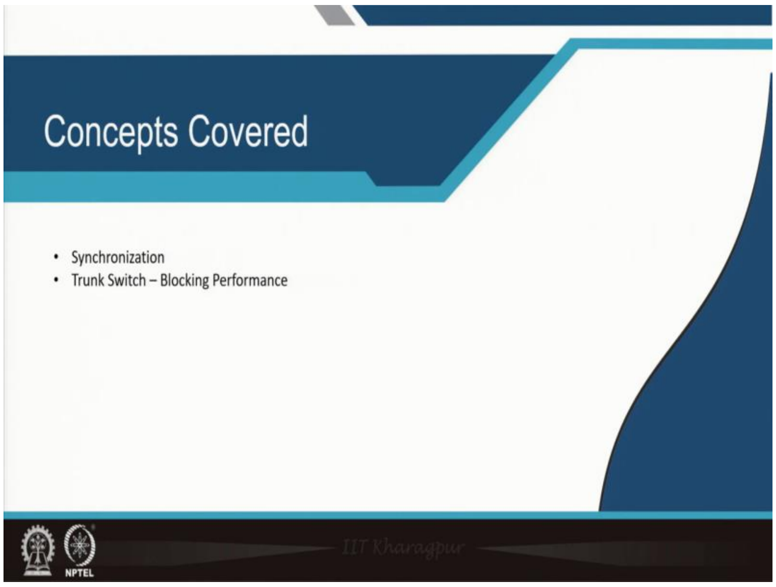
Prof. Goutam Das
G. S. Sanyal School of Telecommunication
Indian Institute of Technology, Kharagpur

Module - 03
Circuit Switched Network
Lecture - 13
Synchronization

Ok, so today is lecture 13, and we have been so far discussing TDM switching and associated circuit switching networks, the concept of this generally which has been employed for voice traffic. So, today we will discuss another new concept which is called Synchronization.

So, we will try to see how we can actually do synchronization and what is actually synchronization that is something we need to discuss first. And then, we will see how to employ synchronization in different elements of this kind of TDM switching or circuit switch networks ok.

(Refer to Slide Time: 01:05)



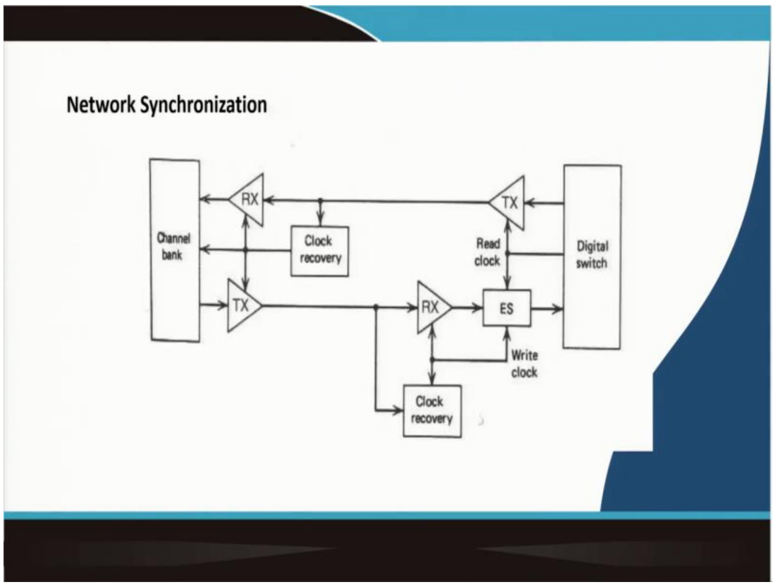
Concepts Covered

- Synchronization
- Trunk Switch – Blocking Performance

So, briefly, if we see today, we will be trying to cover this concept of synchronization that is the first thing, and if time permits, or probably possibly in the next class, we will talk about blocking performance, but for trunk switch. We have so far considered blocking performance of generic switches ok.

Some n to n cross m local switching matrix, but we will also start talking about these trunk switches, and we will try to see how to evaluate blocking performance for that. And for that, we will give a very elementary introduction to queuing theory which is heavily being applied for this trunk switch blocking performance analysis.

(Refer to Slide Time: 01:47)



Network Synchronization

So, that is something we will do. So, let us start talking about network synchronization, ok. So, before that, let me try to understand synchronization. So, what is synchronization? We have point-to-point communication, and we have already talked about synchronization at different levels.

So, you have carrier synchronization in terms of frequency phase you need to synchronize because, at the transmitter, you are probably modulating it with a carrier, and that carrier might have a particular frequency or phase, and then at the receiver side, whoever has done communication probably they will know that receiver site we need to actually synchronize it.

So, that carrier has to be synchronized if it is not synchronized, there are problems in demodulation there will be a lot of problems. So, we have we know these things. I assume that you already know all these things, and then this there is a carrier extraction that is being done; how do you do this synchronization of the carrier? So, basically, you will have a local oscillator that will generate a carrier.

And then you do demodulation with that, but if they are not synchronous, then there is a problem, so that is why, from the incoming signal, you extract the carrier and then synchronize your local oscillator phase and frequency with that. So, this PLL circuit Phase Lock Loop is heavily being used for synchronization. So, that is something we all know about, so this is the carrier.

Then there is also if we are transmitting digital signals, then there is something called clock synchronization, which is also very important. So, you have a particular clock, and then that clock has to be for that bits bit pattern or symbol pattern you have clocks, and then you need to synchronize that clock. So, there are also methods of clock synchronization; whoever has studied digital communication, you are aware of that. So, that is another synchronization.

Now, here what we are talking about is little more than that. So this is called the means you your network synchronization. So, it is all about the in-network operation you are doing, so all those have to be synchronized. So, one of the synchronization we have already given an example when we were talking about this Steven frame. So, frame synchronization is very important because in the frame, if you are doing TDM multiplexing. So, each location of where the bits are there is addressed.

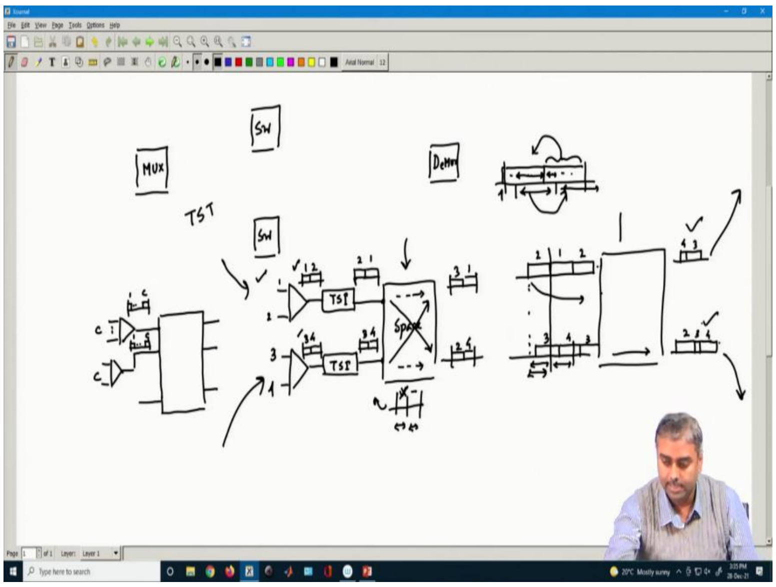
So, you need to synchronize it very nicely; otherwise, you would not know whose data you are because it is a bit stream; you do not know whose data you are putting to which particular port of a switch or in the multiplexer. Also, it will be difficult. So, you need to know exactly how to means locate those timing instructions or information from an incoming bit stream. So, for that, we have already talked about frame synchronization.

So, basically, you construct a frame, and that frame there is some frame identifier, and you actually locate that. So that you always get synchronized with the frame starting point, and from there onwards, you have the whole address of where which particular byte will be for which particular user.

So, these particular things will probably be required. So, this is something we have seen a little bit of frame synchronization that is already we are going ahead from point to point communication towards networking ok. So, in the network, if you see that whatever we have talked about so far, it is a telecom, this kind of switching network, right? So, where does this voice circuitry ok or the voice circuit switch network? So, we transmit voice traffic through that.

So, what we have done we have actually encoded the voice into a PCM-encoded voice, so that is something we have already seen. And then there are some elements in the networking or in the networks that we have constructed.

(Refer to Slide Time: 05:53)



We have already seen that they are like a device called multiplexer, then there are something called switches switch also can be of different means different kinds of switches might be there.

So, there might be this trunk switch; one might be a local subscriber loop switch. So, all kinds of switches might be there, ok. So, whatever it is. So, there are multiplexers, there are switches, and at the end, probably, there will be a demultiplexer, and then it gets connected to the user. So, basically, what happens means first it will come to your voice that will be first digitized.

So, basically, you PCM encode it, and then after that, there will be a multiplexer that takes multiple such voices signals and multiplexes it, and then it goes to the core part of the network where there will be switching elements which just does switching according to your requirement where you want to connect to a particular voice signal wherever you want to connect so that it does ok.

So, sometimes what might happen is there might be a combination of multiplexer demultiplexer in between also. So, you multiplex it to a particular stream, then you start doing switching locally, and after that, it might be further multiplexed to generate a higher data rate, and then it might do some additional switching at that higher level, and then it might be demultiplexed brought down to lower multiplex stream and then further local switching might be done. So, all these combinations might be happening.

But whatever happens, what we have understood is that in a particular network, you have this multiplexer, and you have the switches. Now, we will be talking about the synchronization part of these two devices we need to see, and we will be able to identify also that they require different kinds of multiplexing requirements, and we will also try to see how to provide that multiplexing.

So, let us first try to understand from the perspective of switch what is the requirement of this synchronization. So, generally, what happens to a switch? Suppose I have a switch matrix. So, generally, it will have to be a trunk switch or be it any other switch. So, basically, it will have some input ports and some output ports ok.

So, you have already seen that we might have, let us say, TST kind of switch something like that. So, basically, what will happen? We will already have multiplex data. So, the first portion is time switching, so we do time swapping. So, that kind of switch, or even if it is space switching, is still what might happen. I might get some multiplex data, ok.

So, the multiplex data are coming over here. So, let us say over here some multiplex data are coming. So, basically, some c number of these things are multiplexed ok. So, I have a frame. I have this c unit of data from different voice-encoded signals. So, that comes in every frame, and that fades into the input port of the switch. Similar things will be if the switch inputs are all of a similar kind, then it will all be getting similar kinds of data.

So, let us say in the second port also; there is another c multiplexed voice signal coming over here, so that is also having some c number of data. Now will be accordingly at every time instance; we have already seen that in the TST switch. So, after there might be followed by over here, there might be a time switch over there which will be swapping the things.

Now, in the time switch, there is no problem because it is swapping. So, as long as you have the frame synchronization, that means you know where the frame starts and where everybody's data are located. All those bytes are located; as long as you have this frame synchronization, time swapping is all right; no problem with that.

Now, we will try to see what the problem is in the space part. So, in the space part, what happens suppose I have a time switching followed by space switching, so I have a time switching over here. So TSI time slot interchanger and another one that is also having a TSI ok. So, I have multiple data 1 to c and 1 to c, I have multiplex data that comes over here, and after that, I do time swapping, and accordingly, the data gets swapped over here.

So, let us say; let us say instead of c, if we just put two things; I think that would be a little easier, ok. So, 1 2, 1 2 here some data 1 and 2 ok, let me call this 3 and 4, so this is 3 and 4 ok, now you might do time swapping for them. So, basically, over here, what might happen is this 2's data might come. You might not do time swapping over here 4 comes, ok.

Now, over here, if this multiplex data and this multiplex data if they are not synchronous with each other. So, basically, there is a frame synchronization. Each one of them has frame synchronization. So, basically, in the beginning, you have a bit that specifies the synchronization the way we have discussed frame synchronization. So, it gets synchronized.

Basically, the TSI input gets the synchronization, and accordingly, he knows exactly where the byte location of data is from the one-byte location of data from 2. So, all these things he knows is he can accordingly do the swapping with no problem; no bits are getting means intertwined, or they are not getting misplaced ok.

So, basically, if you have 8 bits, 8 encoded bits, and 8 encoded bits, if you know the beginning of this slot, you will be in when you are swapping, you exactly swap this entire 8-bit and put it over here. So, no problem in that if you do not know the starting point, then probably partially you will start swapping so.

This portion will go to 1 this one the next portion will go to another one. So, then there is a problem because somebody's half data or some partial data is coming, and another guy's partial data is coming, so data is getting jumbled up. So, that is why frame synchronization for times slot interchanger is very important. So, as long as you can do frame synchronization, there is no problem; you can do time swapping.

Now, let us say when I put that to a space switch, these two-time swap data are being faded over here. Now what space switch does space switch? If you remember, it has those slotted things, and at every slot, he does a particular switching. So, it might happen that he has to do bar connectivity in the first slot; that means this data from two must go over here; that is your requirement, and data from 3 must go over here that is your requirement; let us say.

And after that, in the second one, he will be doing. So, this is a cross, and this is a bar; bar means it will be connected like this, so that means data from 1 comes over here, and data from 4 comes over here. This is all ideal things I am talking about, now what will happen? If, at the input, these two things, these two frames are not synchronized to each other, which will generally be happening because this is coming from some other location, this is coming from some other location.

So, from two locations with two different clocks, even if you do clock synchronization,n it might not get synchronized to the actual boundary of frames for the switch. So, if you see it from the perspective of a switch, these 3 and 2 must be completely synchronized. They must be coming at the same location same time instance, and they must be means occupying the time duration for which this has to be switched.

So, that time duration also has to be the same. So, these two additional requirements come into the picture. So, that is very important. Now, as you can see, this is the network synchronization part we are talking about. So, now, just frame synchronization will not help you; you have to also synchronize with respect to this switch every slot. So, switch slot-wise; everything has to be synchronized if there is a slight desynchronization; what might happen, as you can see, I will demonstrate that.

So, basically, let us say the switch input it’s being let us say it is being faded by this. So, 2 followed by 1, and this will be repeated. So, followed by 2, and so on, it goes on like this over here. Let us say I take slight desynchronization, ok. So, over here, let us say 3 followed by 4 followed by 3.

Now, what switch will be doing? So, suppose he synchronizes himself to one of them he can only synchronize to. So, suppose his slot boundary he puts with this guy, so with this guy, he synchronizes it. So, then what will happen? He will be swapping this amount of data 2 will come over here. So, no problem with that because with 2, he has synchronized, but this amount of data will be coming over here, which is not the data from 3. It’s partial data from 3 and partial data from 4.

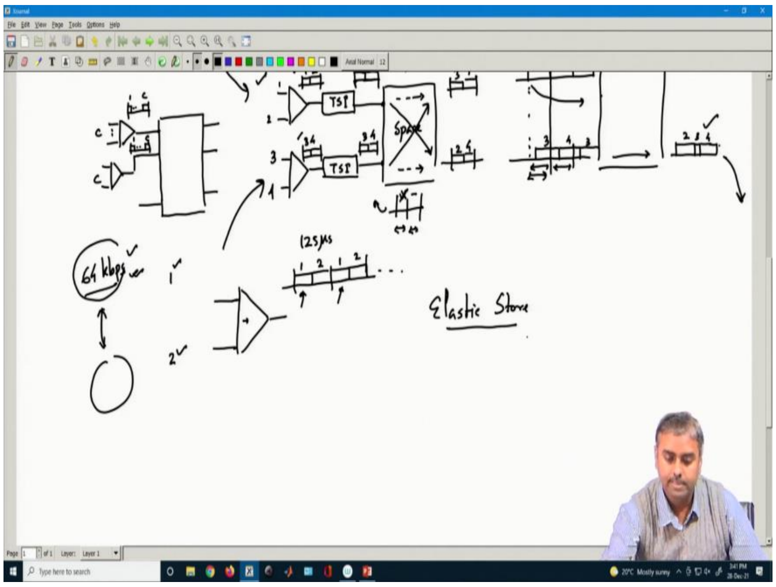
So, now the switching is being done in the wrong fashion, you might be asking, ok, what is the problem? There is no problem as long as even this 4 also goes over here because then the data will be again re-synchronized with respect to the frame, but no guarantee that 4 will be now going over there that is according to the switch requirements. So, 4, according to our example 4, goes over here now what will happen that will be this amount of data.

So, some amount of data from 3 and some amount of data from 4 goes over, and this goes to some other place; now you have no way to synchronize them. The data has been corrupted, so because you have mixed up the data. So, for both the guy’s data for 4 and data for 3, they have been corrupted due to this non synchronization at the switch input and due to the switching that has been done by the space switches. So, this is a typical problem of this kind of network.

So, we will have to basically see how to tackle this problem. So, hopefully, this has explained what the problem is. All we have to now do we have to devise a mechanism to synchronize all the incoming patterns with respect to their frames so that the bytes are exactly synchronous to each other.

So, this is something that we will have to do that is something we need to really perform; we will see how to do that. So, that is something one particular aspect. So, this is the switching where the synchronization is required. Next, we will talk about another thing.

(Refer to Slide Time: 18:30)



So, what is that? That is the multiplexer because I have been told there are two devices; let us say basically, one is a multiplexer one is a switch. Now switch, we have already identified that there is a problem with synchronization, even clock carrier and frame; even if we synchronize, that is not good enough for time switching; it is probably ok, but space switching that is not good enough; we have already identified.

Now, will have to see whether there is a problem with the multiplexer; now, multiplexer, what happens? You have some data. Let us say data from 1. So, it is a digitized voice data of 64 kbps that is coming to another data digitized voice data 64 kbps that is coming, and you are multiplexing them and generating higher data over here multiplexed data.

Now, the problem is this clock is generated at the source, and then it traverses through some terrain ok. So, whatever that might be, depending on the media, it might be wireless media, it might be wired media, whatever it is ok for telephony, generally its wired media.

So, it is going through that, and depending on the transmission characteristics and the source clock, now the source clock is a very important thing. Depending on the source clock, it might not be exactly 64 kbps ok it might be slightly deviating from 64 kbps.

So, it might be 64.0001 kbps or something like that second one might be something little less than 63.999 kbps or something like that. So, there might be a slight deviation very small amount of deviation, and because they are generated from distributed sources that are geographically located at different locations, it is very difficult to synchronize them. You can always generate roughly 64 kbps exactly, but it might not be exactly ok. So, because they might have this drift in the clock, or they might have this difference in these two clocks.

Now, if you try to synchronize them or, let us say, if you try to multiplex them, there will be a problem because what might happen, whichever has a slightly higher data rate over a longer duration of time you will see that it is generating more data and the other one is generating less data. Now what do you do? We have already talked about this multiplexing. It’s either byte interleaving or word interleaving, or bit interleaving.

So, whatever you do, it is actually supposed to be this two-stream 64 kbps you want to interleave. So, it will be one byte from here and one byte from here. Like this, you will be doing keep on 125 microseconds as we have mentioned, one byte from here and one byte from the second, again another 125 microseconds, one byte from here, one byte from here, and so on. It just goes on like this, ok.

Now, the problem is if, over a longer duration, as many bytes being generated by 1 is not equal to as many bytes being generated by 2, then there is a deviation. So, basically, by doing this, you would not be able to take all the data and put it together ok. So, that is the problem which will be happening. So, what will happen? You keep on doing this, so after 4 or 5 frames, you will probably see that ok the one with the higher clock probably has generated more data compared to the lower clock, and that is a problem.

Because now, you have a problem synchronizing the multiplexer because the input fed to the multiplexer you have no control over there input fed is coming from some other distance clock ok. So, it is synchronized to that clock, or it’s generated through that clock. So, that clock synchronization means these two clocks, actually these two distant clocks, you have no way to synchronize them; that is the problem, ok.

So, if you cannot do this synchronization now, you will have difficulty doing the multiplexing in a synchronous manner. So, this is another problem that we will again start discussing how to cope with this particular synchronization problem which will be inherently there; we have to just see how to deal with it ok.

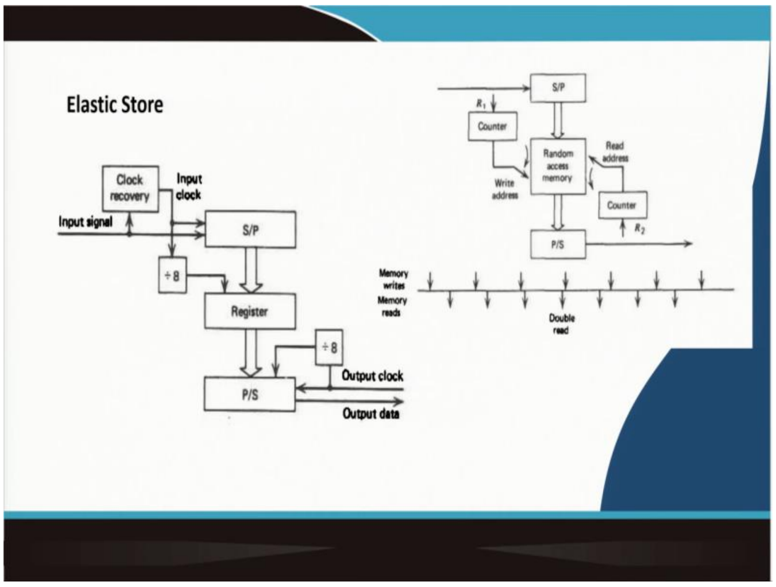
So, that something will have to see because you cannot say that ok; distributedly globally, we will synchronize all the clocks to the precision. So that nobody deviates from here, so that is not possible. So, let us take face the practice or practical scenario that ok, there will be dif means of deviation of the clocks, and I have to see how to cope with that while doing multiplexing. So, the multiplexer has to do something so that we can do this synchronization, and then we have also talked about this switch.

There also there are problems in the switch that we have if especially the space switch; if the inputs input frames are not properly synchronized, then I will have a problem; it has to be synchronized at least to the byte level ok.

Otherwise, we have already seen that partial data from one and partial data from others might go into the actual system, and then the whole data will be jumbled up. So, the voice you will be reconstructed will be a completely wrong voice. It will be an erroneous gibberish voice, ok. So, that is the problem, so that is also something we will have to address.

So, let us try to discuss how we solve each of these problems, the synchronization problem in multiplexers and the synchronization problem in switches, one by one. So, the first problem that we will be discussing or the first mechanism that will be discussing is for the switch, and there is a very nice circuitry it is called the elastic store. This solves the problem of this kind of space switch trying to see what exactly elastic store does ok.

(Refer to Slide Time: 24:37)



So, this is the circuitry of an elastic store; an elastic store is a very simple circuit. It is just having one serial-to-parallel converter followed by one parallel-to-serial converter, so serial-to-parallel converter, sorry. So, one serial-to-parallel converter followed by one parallel-to-serial converter, and you have a register in between, as you can see ok.

And, of course, this is the input stream that is coming in ok from outside, and this is the part where it is actually getting into the switch. So, before entering into the switch, you are devising some kind of processing so that you do not have these kind of problems ok this kind of byte level problems that you are having.

So, this is exactly what you are trying to do, ok. So, you are trying to synchronize them in terms of byte level, ok. So, what you actually do over here, let us try to understand. So, basically, this serial to parallel converter it is a kind of shift register ok.

So, what happens if every bit comes? It shifts one by one, and generally, it is a; it is a byte long. So, 8 bit, and as you can see, the clock also for the register that is also of size 8 means you divide the clock with means by 8 times and then. So, your clock now becomes a byte-level clock. Sorry yes, the byte-level clock or the slot-level clock ok.

So, basically, what you do is input signal that comes from there you recover the clock ok. So, this is the bit-level clock; you take that bit-level clock and divide it by 8 now you get the byte-level clock ok. So, that means the slot boundaries of the clock will be exactly synchronized to the slot boundaries; we will see how that slot boundaries are being defined now the data are coming. So, when the data are coming, they are faded to this shift register ok.

So, in the shift register, one by one, data will be coming with the clock; this clock is faded from the actual recovered clock from the input signal. So, basically, there will be one after one, they will be faded to the shift register, and like this, 8-bit will be transferred in this shift serial to the parallel converter or shift register; after that, there will be a byte boundary which will be enabling this register. So, basically, all the data from this shift register will come to this register this particular register fine.

And now, what will be happening? So, basically, a byte is being stored over here in this register, and then the next 8-bit data again, similarly with the serial to parallel converter, will be stored over this serial, parallel converter register ok. Now this is the writing part of the register; the bottom part is the reading part. What you are trying to do is something like this. So, when you have this data incoming, you are also trying to actually take the clock; this is the clock that you. Are this is the clock that you are taking from the switch itself?

That clock, again, you divide by 8, and similarly, you actually store it in means this storage is a different thing; it is a parallel to serial converter. So, basically, that reads the data from the register. So, what you do, this clock, you again divide by 8, which means, again, byte boundary you create, but that is with respect to the switch itself ok.

So, you create that bound byte boundary, and whenever the switch requires a particular byte, it will be reading from the register. So, there is a kind of delay that you are creating with respect to the incoming data. So, incoming data you are taking and storing in a particular shift register, and then you are reading it with respect to that. So, one by one, you are reading.

So, everything you are converting to byte, whatever incoming data is coming, you are converting to byte and then reading it to the register; you are doing that while means or sorry writing it to the register and then when the data has to be read that is by the method or by the clock of the switch.

The switch is also creating a byte boundary, and from there, you are enabling this data to be read whenever the switch is ready for doing another byte switching. So, basically, all you are trying to do is you are while means manipulating the shift register or manipulating this register so it is being fed by data that is incoming and read by data which is at the switch itself.

So, we will demonstrate in the next class in a very nice fashion how this very simple device actually completely gives you a very nice; means boundary of those slots, and the data is read very nicely at the boundary only; it does not really take some portion of data or partial switching it does not do we will try to demonstrate that part with this very nice device ok.

So, this is the elastic store. We will talk about this elastic store a little bit more, and then we will try to see how to improve that, and then finally, we will come to the multiplexer synchronization that we were talking about later on ok.

Thank you.